## REMARKS

Claims 1-15 were pending as of the action mailed on August 10, 2007. Claims 1, 9, 12 and 15 are in independent form.

Claims 1, 9, 12 and 15 are being amended. Claims 2 and 13 are being cancelled. No new matter has been added. Support for the amendments can be found in the specification at least in paragraphs 0006-0008, 0026-0030, 0032, and 0036-0038. Paragraphs 0012 and 0031 have been amended to corrected clerical errors.

Reexamination of the application and reconsideration of the action are respectfully requested in light of the foregoing amendments and the following remarks.

### Objection to Specification

The applicant has amended the specification to address the examiner's objection to paragraph 0012 and remove duplicate sentences from the paragraph.

### Section 101

Claims 1-15 were rejected under 35 U.S.C. § 101 as allegedly directed to non-statutory subject matter.

The applicant has amended independent claims 1 and 9, removing the offending reference to an information carrier. Claims 2-8 and 10-11 depend from the foregoing independent claims.

For the foregoing reasons, the applicant respectfully submits that the rejection of claims 1-11 should be withdrawn.

The applicant respectfully traverses the rejection of claims 12-15. The limitations of these claims are recited in means plus function form. "Where means plus function language is used to define the characteristics of a machine or manufacture invention, such language must be interpreted to read on only the structures or materials disclosed in the specification and "equivalents thereof" that correspond to the recited function. Two *en banc* decisions of the Federal Circuit have made clear that the USPTO is to interpret means plus function language according to 35 U.S.C. § 112, sixth paragraph." MPEP § 2106 page 2100-7 (Rev. 6, Sept. 2007) (emphasis added)

Because the claims' limitations must be interpreted to read only on structures or materials, they necessarily satisfy § 101.

For the foregoing reason, the applicant respectfully submits that the rejection of claims 12-15 should be withdrawn.

## Section 102

Claims 1-15 were rejected under 35 U.S.C. § 102(e) as allegedly anticipated by U.S. Publication No. 2002/0083073 ("Vaidya").

Vaidya is not relevant art.

Vaidya describes systems and methods for use by users in "managing a layered hierarchical data set." [Vaidya Abstract] "Embodiments of the invention are for managing a layered hierarchical data set." [Vaidya ¶ 0044] "When a user accesses data element in the hierarchical data set, a sub-tree of the hierarchical data is presented to the user. When the user operates on a data element, the state attribute of the node associated with the data element is updated. Using the updated state attribute, the entire hierarchical data set is managed so that when a user accesses the data again, it is presented correctly." [Vaidya Abstract, ¶ 0029]

It is important to recognize that Vaidya relates to the management of data that resides in a server and can be managed by multiple users. In fact, the hierarchical data set that is being managed typically resides in a backend database:

> With reference to embodiments of the present invention, hierarchically organized information 950 typically resides in the back tier in a database 955, while the data is typically accessed by client computer 901. In operation, client computer 901 makes a request 960, to the database 955 for the hierarchically organized information 950. The request 960 is then used to present the user with an appropriate tree by accessing the database 955 and locating the proper nodes in the hierarchical information 960 to re-create a sub-tree 965 on the client computer 901. [Vaidya ¶ 0093]

A user can access and operate on data in the layered hierarchical data set. When the user "accesses data in the hierarchical data set, a subset of the hierarchical data set (i.e., a sub-tree) is presented to the user." If the user does operate on the data, the data and its state attributes are modified. [Vaidya ¶ 0047] "In operation, client computer makes a request, to the database for the hierarchically organized information. The request is then used to present the user with an appropriate tree by accessing the database and locating the proper nodes in the hierarchical information to re-create a sub-tree on the client computer." [Vaidya ¶ 0093; reference numbers omitted]

The hierarchical data set can be "cached in a server as a DOM tree." [Vaidya ¶ 0065, FIG. 2; emphasis added] It is important to recognize that such a DOM is similar in name only to the DOM built and used by a Web browser in a client, to which the present application relates.

"Typically, when a user asks for data, such as that found in a server with a hierarchical layout like that shown in FIG. 2 [the data cached in a server as a DOM tree], the user receives a merged view of the data . . . which is what a user would operate on . . . ." [Vaidya ¶ 0067, *see also* ¶ 0093 quoted above]

"When an user asks for the data, the user receives a merged tree view of the data. Therefore, the server has the responsibility of converging data into a merged tree, and converting the merged tree transactions to the actual tree on the server." [Vaidya ¶ 0033]

Vaidya describes an updateNode function that performs this responsibility of converging data. The updateNode function "converts the merged tree transactions to the actual tree on the server." [Vaidya ¶ 0070; emphasis added] The updateNode function is a function that can reside on a server. [Vaidya ¶ 0070] It performs a number of functions on the user's sub-tree, i.e., on the merged view of the hierarchical data set that the user computer received when the user asked for data. [Vaidya ¶¶ 0067, 0078, 0093]

Claim 1. In rejecting claim 1, the examiner stated that:

> Vaidya teaches . . . :
>
> • Identifying a change of a user interface (UI) element that references a node of a document object model (DOM) hierarchy (Vaidya Para 0047-0048 and Figure 1) Vaidya teaches that a user operation on a DOM tree node is identified . . . .

The applicant respectfully disagrees. Vaidya does not describe or relate to UI elements that references a DOM. In the light of the specification, the claim language clearly refers to user interface elements in a user interface page being presented to a user by a web browser that maintains a DOM of the presentation. [*See*, e.*g*., ¶¶ 0003, 0005, 0007, 0008; emphasis added] Nevertheless, to forestall any further unwarranted broad reading of the claim language, the applicant has amended the language to make this clear. However, this amendment should not require a further search on the examiner's part, because the examiner will have conducted a thorough search, as set forth in MPEP § 904.03 (8th Ed., Rev. 6), and covered "all subject matter which the examiner reasonably anticipates might be incorporated into applicant's amendment."

Turning to the actual disclosure that the examiner relies on, paragraphs 0047 and 0048 of Vaidya reads as follows.

> [0047] Referring again to FIG. 1, at operation 110 a user accesses data in the hierarchical data set. In response, a subset of the hierarchical data set (i.e., a sub-tree) is presented to the user at operation 120. Next, it is determined if the user has operated on a subset of the data at operation 130. If not, the process repeats at operation 120. Otherwise, the data and state attributes are modified at operation 140.
>
> [0048] When a user again needs access to the hierarchical information, one embodiment of the present invention shown in FIG. 7 presents it to the user in the following manner. At operation 700, the state attributes in the tree are traversed so that the appropriate information (i.e., nodes) may be presented to the user. Next, at operation 710 it is determined whether the present state attribute has a default value. If the state attribute does in fact have a default value then it has not been changed. Thus, the process repeats and other state attribute are examined at operations 700 and 710 to find one that is non-default.

These paragraphs do not in fact describe user operations on a DOM tree. User operations on a DOM tree – more accurately, on a merged partial tree presented to a user – are described elsewhere in Vaidya, as noted above, for example in ¶¶ 0065, 0067 and 0093. And, as noted above, those operations do not relate to user interface elements, but rather to direct user manipulation of the hierarchical data set, which is cached on a server by a DOM tree, in those embodiments of Vaidya that use a DOM tree.

For the foregoing reasons, the applicant respectfully submits that the rejection of claim 1 should be withdrawn.

In rejecting claim 1, the examiner further stated that:

> Vaidya teaches . . . :
>
> • Determining whether the change of the UI element can be applied to the DOM hierarchy by using an update function (Vaidya See Para 0069-0076)

The applicant respectfully disagrees.

The portion of Vaidya relied on by the examiner reads as follows.

> [0069] updateNode
>
> [0070] One embodiment of the present invention converts the merged tree transactions to the actual tree on the server. In one embodiment, the function responsible for this is called updateNode, which may reside on a server, such as a Registry server. This function performs a variety of operations on the client side. These operations include, for instance:
>
> [0071] Override value: change the default value to a new overridden value, or change an already overridden value to a new overridden value.
>
> [0072] Delete: remove a default node, or delete a user defined node (one that does not exist in the default tree).
>
> [0073] Reset: reset the node to a default value.
>
> [0074] Add: add a new node to the user's sub-tree.
>
> [0075] Replace: replace a default node with a new node with either the same name, or different name to the user's sub-tree.
>
> [0076] FIG. 4 is a partial view of the DOM tree of FIGS. 2 and 3 where updateNode function has been used to change the Font layer 400 from Times 292 (in both FIGS. 2 and 3) to Arial 405, and the Size node 410 under Lines 415 has been changed from 12 in FIG. 2 to 10 in FIG. 3 to 14 in FIG. 4.

Nothing in these paragraphs describes or suggests determining whether an update function can be used to make the identified change in the DOM. Nothing in Vaidya suggests anything other than that the described updateNode function can perform all the operations required of it; and no mention is made of any changes that can be made, but not made by an update function. The rejection under § 102 is therefore improper, because the examiner has failed to show that Vaidya discloses "determining whether," as explicitly recited in the claim.

For the foregoing additional reasons, the applicant respectfully submits that the rejection of claim 1 should be withdrawn.

In rejecting claim 1, the examiner finally stated that:

> Vaidya teaches . . . :
>
> • Else, setting a dirty flag. (See Para 0062) Vaidya teaches that the replaced state attribute for a given node was not known to the previous layers and is new and will have to be updated in the cache.

The applicant respectfully disagrees. Preliminarily, the examiner does not – and could not – state that Vaidya actually teaches setting a dirty flag, which is what the claim recites. For at least this reason, a rejection under § 102 is improper and should be withdrawn.

The teaching the examiner does find in Vaidya is that the addition of a node into the hierarchical data set by a user causes that node to have the state "replaced." In fact, all that the portion of Vaidya relied on by the examiner states is the following:

> [0062] Replaced: indicates that this node is not defined in the previous layers. The owner of the current layer has created this node, and another node with the same name may exist in another layer, but is replaced by this new node.

Furthermore, the claim recites that setting the dirty flag occurs as an alternative to applying an update function, i.e., when no update function is applied. Contrary to that, in Vaidya the updateNode function is always applied and, in fact, it is this function that changes the state to "replaced." [Vaidya ¶ 0077 and FIG. 5] Therefore, the alternative operation recited in the claim is not disclosed by Vaidya, and the rejection under § 102 is improper.

For the foregoing reasons, the applicant respectfully submits that the rejection of claim 1 should be withdrawn.

Claim 3. In rejecting claim 9, the examiner stated that:

> Vaidya teaches the product wherein the dirty flag indicates invalidity of a cached DOM representation of the UI element (See Para 0060)

The applicant respectfully disagrees. Paragraph 0060 reads in its entirety as follows:

> [0060] Modified: indicates that the node in the current layer has a value different from the default value.

Nothing in this paragraph refers to a cached representation of part of a DOM, as recited in the claim. (Recall that, as explained above in reference to paragraph 0065, the DOM of Vaidya is itself entirely a cache on a server of the hierarchical data set that is being managed by a user.)

Moreover, that a node in a tree has a value that differs from a default value does not mean that the node's value is invalid.

In rejecting claim 3, the examiner further stated that:

> Vaidya teaches the state of the node has changed and therefore the cache and the tree are not in sync and needs to be updated (See also Para 0057-0063).

The applicant respectfully disagrees.

First, as already mentioned, the cache in Vaidya is the DOM, so they are necessarily in sync. However, if by "the tree" the examiner is referring to the hierarchical data set that is being managed, then "the tree" of Vaidya is not in a Web browser in a client computer, as recited in the claim, and so does not correspond to anything recited in the claim.

Second, the paragraphs of Vaidya cited by the examiner do not support the examiner's position that "the cache and the tree are not in sync and needs to be updated." They read as follows:

> [0057] According to one or more embodiments of the present invention state attributes are added to each node in the hierarchy, for instance in the manner shown in FIG. 8. The state attribute has information relating to the last operation that was performed on its corresponding data element.

> [0058] Possible values of the state attribute include:
>
> [0059] Default: indicates that this node has come from the previous layer or the system default layer. This is the default value of the attribute.
>
> [0060] Modified: indicates that the node in the current layer has a value different from the default value.
>
> [0061] Deleted: indicates from the current layer's perspective that this node and all its sub-nodes are deleted (data does not exist).
>
> [0062] Replaced: indicates that this node is not defined in the previous layers. The owner of the current layer has created this node, and another node with the same name may exist in another layer, but is replaced by this new node.
>
> [0063] Depending on the value in the state attribute, different merged DOM trees are presented to the user. One example is shown in FIGS. 2-4.

It does not appear that anything in these paragraphs relates to data not being in sync and requiring an update.

For the foregoing reasons, the applicant respectfully submits that the rejection of claim 3 should be withdrawn.

Claim 4. In rejecting claim 4, the examiner stated that:

> Vaidya teaches the product wherein the dirty flag indicates invalidity of a cached DOM representation of a further UI element that is a parent of the UI element.

The examiner cited nothing in Vaidya to support this assertion.

For at least the foregoing reason, the applicant respectfully submits that the rejection of claim 4 should be withdrawn.

Claims 7 and 8. In rejecting claims 7 and 8, the examiner stated that:

> Vaidya teaches the product wherein the cached DOM representation comprises a sub-hierarchy of the DOM hierarchy and the node is the root node of the sub-hierarchy and the at least one attribute is an attribute of a sub-node of the sub-hierarchy (See Para 0047-0048 and 0073-0075).

The applicant respectfully disagrees.

As explained earlier, paragraphs 0047-0048 do not, in fact, describe user operations on a DOM tree. User operations on a DOM tree – more accurately, on a merged partial tree presented to a user – are described elsewhere in Vaidya, as noted above, for example in ¶¶ 0065, 0067 and 0093. And, as noted above, those operations do not relate to user interface elements, but rather to direct user manipulation of the hierarchical data set, which is cached on a server by a DOM tree, in those embodiments of Vaidya that use a DOM tree.

Also, as noted above, paragraphs 0073-0075, which are reproduced below, do not teach a cached sub-hierarchy in a client computer that is a cached representation of a UI element that has experienced a change that was identified in a Web browser in the client computer, as recited in the claim. All these paragraphs describe is the addition, replacement, or resetting of a node to a tree:

> [0073] Reset: reset the node to a default value.
>
> [0074] Add: add a new node to the user's sub-tree.
>
> [0075] Replace: replace a default node with a new node with either the same name, or different name to the user's sub-tree.

For the foregoing reasons, the applicant respectfully submits that the rejection of claims 7 and 8 should be withdrawn.

Claim 9. In rejecting claim 9, the examiner stated that:

> Vaidya teaches . . . :
>
> • receiving a request to render a user interface (UI) element of a UI-tree (See Figure 2) Vaidya teaches the system renders the UI tree.

The applicant respectfully disagrees.

Vaidya does not teach rendering a user interface element of a user interface page that has a hierarchical structure represented by the recited UI tree in the client computer.

For the foregoing reasons, the applicant respectfully submits that the rejection of claim 9 should be withdrawn.

In rejecting claim 9, the examiner further stated that:

> Vaidya teaches . . . :
>
> • determining whether a dirty flag is set for a cached document object model (DOM) representation of a flagged UI element that is on the path of the UI element in the UI tree (See Para 0058-0063) Vaidya teaches the setting of a state value that signifies that the value of the node has changed in the tree and starts a process to update the cache on the server.

The applicant respectfully disagrees.

The paragraphs the examiner cites, as well as the preceding paragraph that sets the context, read as follows.

> [0057] According to one or more embodiments of the present invention state attributes are added to each node in the hierarchy, for instance in the manner shown in FIG. 8. The state attribute has information relating to the last operation that was performed on its corresponding data element.
>
> [0058] Possible values of the state attribute include:
>
> [0059] Default: indicates that this node has come from the previous layer or the system default layer. This is the default value of the attribute.
>
> [0060] Modified: indicates that the node in the current layer has a value different from the default value.
>
> [0061] Deleted: indicates from the current layer's perspective that this node and all its sub-nodes are deleted (data does not exist).
>
> [0062] Replaced: indicates that this node is not defined in the previous layers. The owner of the current layer has created this node, and another node with the same name may exist in another layer, but is replaced by this new node.
>
> [0063] Depending on the value in the state attribute, different merged DOM trees are presented to the user. One example is shown in FIGS. 2-4.

Nothing in the foregoing paragraphs states or suggests that a dirty flag is set for "a cached document object model (DOM) representation of a flagged UI element," as recited in the claim. In addition, although the examiner refers to "a process to update the cache on the server," the applicant sees nothing in the cited paragraphs that refers to such a process or such a cache. Moreover, in light of the specification, under the broadest reasonable interpretation of the claim, it would have been clear that the UI element being rendered is part of a UI page represented by the recited UI tree in a client, that the cached DOM representation resides in the client, that the flagged UI element is not the same element as the UI that is the subject of the request to be rendered, and that the new DOM representation is generated in the client and not in some server. Each of these features has now been added to the claim.

In rejecting claim 9, the examiner further stated that:

> Vaidya teaches . . . :
>
> • The flagged UI element having a reference to a node of the DOM hierarchy (See Para 0065). Vaidya teaches that each node in the DOM is a UI element displayed to the user.

The applicant respectfully disagrees.

Each node in the DOM of Vaidya is a node on a server; and the DOM, as explained above, represents a layered hierarchical data set that is managed by the user. The nodes are not nodes of a UI tree that represents a user interface page. Paragraph 0065 teaches that the hierarchical data set can be "cached in a server as a DOM tree." This does not support the examiner's assert that "each node in the DOM is a UI element displayed to the user."

For the foregoing reasons, the applicant respectfully submits that the rejection of claim 9 should be withdrawn.

In rejecting claim 9, the examiner further stated that:

> Vaidya teaches . . . :
>
> • if the dirty flag is set, generating a new DOM representation of the flagged UI element and inserting the new DOM representation into the DOM hierarchy by using the UI-tree and the reference to the node (See Para 0047-0048) Vaidya teaches the presentation of a new sub-tree when the user adds information and the new representation reflects the change. The state attribute is set that indicates that the value is no longer current with the cached structure of the DOM on the server but allows the display to be updated

The applicant respectfully disagrees. The paragraphs of Vaidya relied on by the examiner, which are reproduced below, do not actually provide the teachings that the examiner asserts are present.

> [0047] Referring again to FIG. 1, at operation 110 a user accesses data in the hierarchical data set. In response, a subset of the hierarchical data set (i.e., a sub-tree) is presented to the user at operation 120. Next, it is determined if the user has operated on a subset of the data at operation 130. If not, the process repeats at operation 120. Otherwise, the data and state attributes are modified at operation 140.
>
> [0048] When a user again needs access to the hierarchical information, one embodiment of the present invention shown in FIG. 7 presents it to the user in the following manner. At operation 700, the state attributes in the tree are traversed so that the appropriate information (i.e., nodes) may be presented to the user. Next, at operation 710 it is determined whether the present state attribute has a default value. If the state attribute does in fact have a default value then it has not been changed. Thus, the process repeats and other state attribute are examined at operations 700 and 710 to find one that is non-default.

These paragraphs teach that the user's operations on a sub-tree of the hierarchical data set can modify data and state attributes. [¶ 0047] None of this teaches or suggests "inserting the new DOM representation into the DOM hierarchy," as stated by the examiner and as recited by the claim.

The difference is made clear by the subsequent paragraph 0049, which reads as follows:

> [0049] When a non-default state attribute is encountered, the user is presented with a subtree at operation 720 where the data element associated with the non-default state attribute and all of its parents up to the root are presented as a sub-tree. If the user again changes a data element, then the process repeats where the associated state attribute is updated and such an update is propagated to the complete tree residing on the server.

What this teaches is that the presence of a non-default state determines what is presented to the user as a sub-tree. It does not teach what is recited in the claim, namely, that depending on whether the dirty flag is set, either a new DOM representation is generated in the client of the flagged element and inserted into the DOM hierarchy, or a cached DOM representation in the client of the flagged element is inserted into the DOM hierarchy. Furthermore, nothing in the cited paragraphs teaches that in either case, the insertion is done using the UI tree that represents the user interface page and a reference to a node of the DOM hierarchy that the flagged element has.

For the foregoing reasons, the applicant respectfully submits that the rejection of claim 9 should be withdrawn.

In rejecting claim 9, the examiner finally stated that:

> Vaidya teaches . . . :
>
> • and else, inserting the cached DOM representation into the DOM hierarchy by using the UI-tree and the reference to the node (See Para 0078 and 0085) If the state attribute is set to default then the cached tree on the server will be rendered on the display.

The applicant respectfully disagrees.

The examiner relies on Vaidya as teaching: "If the state attribute is set to default then the cached tree on the server will be rendered on the display." Even if this were an accurate description of Vaidya, it does not meet the limitations of the claim, because it is clear that the cached representation of the claim is in the client and not in the server.

Moreover, as the handling of default nodes, paragraph 0078 states that "if the state attribute associated with the node is default on server, the node is deleted from the user's sub-tree." Contrary to the examiner's characterization, this does not state that "the cached tree on the server will be rendered on the display." The other mention of the default state provides: "On the other hand, if the node does not exist (user was using the default value up to now), it is deleted using the addNode operation with the node's state attribute value set to 'deleted'." (Emphasis added) This also does not teach that "the cached tree on the server will be rendered on the display."

The other paragraph on which the examiner relies reads as follows.

> [0085] At operation 602 the call to the sub-tree is checked for success. If the call is a failure, then at operation 603 an appropriate error message is

> sent back (and optionally logged in an error log) along with the current value of the node for the nodePath in the DOM tree.

The applicant is at a loss to understand the relevance of this paragraph to any limitation of the claim.

For the foregoing reasons, the applicant respectfully submits that the rejection of claim 9 should be withdrawn.

Claim 12. Independent claim 12 was rejected "along the same rationale" as claim 1.

For the reasons set forth above in reference to claim 1, the applicant respectfully submits that the rejection of claim 12 should be withdrawn.

Claim 15. Independent claim 15 was rejected "along the same rationale" as claim 9.

For the reasons set forth above in reference to claim 9, the applicant respectfully submits that the rejection of claim 15 should be withdrawn.

Remaining claims. The remaining claims depend from the claims that have been addressed, and the rejection the remaining should be withdrawn for at least the reasons set forth in reference to the respective independent claims or corresponding dependent claims.

## Conclusion

For the foregoing reasons, the applicant submits that all the claims are in condition for allowance.

By responding in the foregoing remarks only to particular positions taken by the examiner, the applicant does not acquiesce with other positions that have not been explicitly addressed. In addition, the applicant's selecting some particular arguments for the patentability of a claim should not be understood as implying that no other reasons for the patentability of that claim exist. Finally, the applicant's decision to amend or cancel any claim should not be understood as implying that the applicant agrees with any positions taken by the examiner with respect to that claim or other claims.

Please apply any charges not otherwise paid or any credits to deposit account 06-1050.

Respectfully submitted,

Date: 12 Nov 07

/Hans R. Troesch/
Hans R. Troesch
Reg. No. 36,950

**Customer No. 32864**
Fish & Richardson P.C.
Telephone: (650) 839-5070
Facsimile: (650) 839-5071

50447683.doc